Clustered Capacitated Vehicle Routing File Format

M. Sevaux[1,2]

[1]Helmut-Schmidt-Universität, Logistics Management Department, Hamburg, Germany
[2]Université de Bretagne-Sud, Lab-STICC, Lorient, France

June 2011

Abstract

This document presents a simple extension of the TSPLIB file format to serve our needs in the Clustered Vehicle Routing Problem types. Instead of creating a new file format or putting ASCII files online with a simple description, we have chosen to extend the TSPLIB file format.

This document is an extension of the TSPLIB file format description proposed in [1] to be used for the Inventory Routing Problem and variants. No repetition of the information in [1] is given, unless strictly necessary, but the present file is self-contained to describe our instances.

TSPLIB is a library of sample instances for the TSP (and related problems) from various sources and of various types. In addition to the following problem classes TSP, HCP, ATSP, SOP and CVRP described in [1], we have created a new class called:

Clustered capacitated vehicle routing problem (CCVRP)

We are given $n-1$ nodes, one depot and distances from the nodes to the depot, as well as between nodes. All nodes have demands which can be satisfied by the depot. For delivery to the nodes, trucks with identical capacities are available. Each node belongs to a cluster $r \in \{1, \cdots, R\}$ (cluster 0 is reserved for the depot). In addition to the Classical CVRP, a cluster should be visited by only one truck and this visit cannot be split (once a truck starts to visit a cluster, it should complete the visit of all nodes in that cluster). The problem is to find tours for the trucks of minimal total length that satisfy the node demand without violating the truck capacity constraint and the clustering constraints. The number of trucks is not specified. Each tour visits a subset of the nodes and starts and terminates at the depot.

1 The file format

Each file consists of a **specification part** and a **data part**. The specification part contains information on the file format and on its contents. The data part contains explicit data.

1.1 The specification part

All entries in this section are of the form <*keyword*> : <*value*>, where <*keyword*> denotes an alphanumerical keyword and <*value*> denotes alphanumerical or numerical data. The terms <*string*>, <*integer*> and <*real*> denote character string, integer or real data, respectively. The order of specification of the keywords in the data file is arbitrary (in principle), but must be consistent, i. e., whenever a keyword is specified, all necessary information for the correct interpretation of the keyword has to be known. Below we give a list of all available keywords.

1.1.1 `NAME` : <*string*>

Identifies the data file.

1.1.2 `TYPE` : <*string*>

Specifies the type of the data. Possible types are

- `CVRP` Capacitated vehicle routing problem data
- `CCVRP` Clustered capacitated vehicle routing problem data

1.1.3 `COMMENT` : <*string*>

Additional comments (usually the name of the contributor or creator of the problem instance is given here).

1.1.4 `DIMENSION` : <*integer*>

For a TSP or ATSP, the dimension is the number of its nodes. For a CVRP or CCVRP, it is the total number of nodes and depots. For a TOUR file it is the dimension of the corresponding problem.

1.1.5 `CAPACITY` : <*integer*>

Specifies the truck capacity in a CVRP or CCVRP.

1.1.6 `TOUR_LENGTH` : <*integer*>

Specifies the maximum tour length of any truck in a CVRP or CCVRP.

1.1.7 EDGE_WEIGHT_TYPE : <*string*>

Specifies how the edge weights (or distances) are given. The values are

EUC_2D_INT Weights are Euclidean distances in 2-D (integer rounding)

EUC_2D_DBL Weights are Euclidean distances in 2-D (real value)

EUC_2D_1DD Weights are Euclidean distances in 2-D (real value but rounded with 1 decimal digits)

Note that the original EUC_2D has been replaced by the EUC_2D_INT.

Up to now we have used only the EUC_2_INT specification which is the most commonly used for CVRP and CCVRP problem types.

Some of the sections used in the original paper are not relevant if we limited our usage to the EUC_2D_* specification. We refer the reader to the original paper [1] for a complete description of other sections.

1.1.8 NODE_COORD_TYPE : <*string*>

Specifies whether coordinates are associated with each node (which, for example may be used for either graphical display or distance computations). The values are

TWOD_COORDS Nodes are specified by coordinates in 2-D

NO_COORDS The nodes do not have associated coordinates

Since the default value is NO_COORDS, it is necessary to choose the right option here, especially for the EDGE_WEIGHT_TYPE specification compatibility.

1.1.9 EOF :

Terminates the input data. This entry is optional.

1.2 The data part

Depending on the choice of specifications some additional data may be required. This data is given in corresponding data sections following the specification part. Each data section begins with the corresponding keyword. The length of the section is either implicitly known from the format specification, or the section is terminated by an appropriate end-of-section identifier.

1.2.1 `NODE_COORD_SECTION` :

Node coordinates are given in this section. Each line is of the form

<*integer*> <*real*> <*real*>

if `NODE_COORD_TYPE` is `TWOD_COORDS`. Note that for some instances, coordinates are integer. Think about automatic conversion while reading the file.

The integers give the number of the respective nodes. The real numbers give the associated x and y coordinates.

1.2.2 `DEPOT_SECTION` :

Contains a list of possible alternate depot nodes. This list is terminated by a -1.

1.2.3 `DEMAND_SECTION` :

The demands of all nodes of a CVRP and CCVRP are given in the form (per line)

<*integer*> <*integer*>

The first integer specifies a node number, the second the demand. The depot nodes should not occur in this section since their demand is null for all periods.

1.2.4 `CLUSTER_SECTION` :

The cluster information of all nodes of a CCVRP are given in the form (per line)

<*integer*> <*integer*>

The first integer specifies a node number, the second the cluster identifier. The depot nodes should not occur in this section since their cluster identifier is always 0.

2 The distance functions

For the various choices of `EGDE_WEIGHT_TYPE`, we now describe the computations of the respective distances. In each case we give a (simplified) C-implementation for computing the distances from the input coordinates.

In some cases, distances are required to be integral. In that case, we round to the nearest integer (as rounding is defined mathematically – if $\geq x.5$ rounds up to $x+1$, otherwise rounds down to x). Below we have named the rounding function "`nint`".

2.1 Euclidean distance (L_2-metric)

For edge weight type `EUC_2D_INT`, floating point coordinates must be specified for each node. Let `x[i]` and `y[i]` be the coordinates of node i.

In the 2-dimensional case `EUC_2D_INT`, the distance between two points i and j is computed as follows:

```
xd = x[i] - x[j];
yd = y[i] - y[j];
dij = nint( sqrt( xd*xd + yd*yd) )
```

For edge weight type `EUC_2D_DBL`, the distance between two points i and j is computed as follows:

```
xd = x[i] - x[j];
yd = y[i] - y[j];
dij = sqrt( xd*xd + yd*yd)
```

For edge weight type `EUC_2D_1DD`, the distance between two points i and j is computed as follows:

```
xd = x[i] - x[j];
yd = y[i] - y[j];
dij = 1/10*floor(10*sqrt( xd*xd + yd*yd))
```

This distance measure has been used in [2] but the user should be aware that this function really depends on the coordinate values.

The function `sqrt` is the C square root function and `floor` is the classical mathematical floor function that rounds down real numbers to the integer below.

3 Access

CCVRP instances are available at `http://www.univ-ubs.fr/or/`

References

[1] G. Reinelt. Tsplib95. Internal report, Universität Heidelberg, Heidelberg, Germany, 1995. `http://neo.lcc.uma.es/radi-aeb/WebVRP/data/Doc.ps`.

[2] N. Kohl, J. Desrosiers, O.B.G. Madsen, M.M. Solomon, and F. Soumis. 2-path cuts for the vehicle routing problem with time windows. *Transportation Science*, 33(1):101–116, 1999.